Міністерство освіти і науки України
Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Факультет інформатики та обчислювальної техніки
Кафедра обчислювальної техніки

Лабораторна робота №2

з дисципліни

«Алгоритми та структури даних»

Виконав: студент групи ІМ-42

Максим Крамаренко Юрійович

номер варіанту: 17

Перевірив:

Сергієнко А. М.

Київ 2025

Постановка задачі

1. Створити список з n (n > 0) елементів (n вводиться з клавіатури), якщо інша кількість елементів не вказана у конкретному завданні за варіантом.
2. Тип ключів (інформаційних полів) задано за варіантом.
3. Вид списку (черга, стек, дек, прямий однозв’язний лінійний список, обернений однозв’язний лінійний список, двозв’язний лінійний список, однозв’язний кільцевий список, двозв’язний кільцевий список) вибрати самостійно з метою найбільш доцільного розв’язку поставленої за варіантом задачі.
4. Створити функції (або процедури) для роботи зі списком (для створення, обробки, додавання чи видалення елементів, виводу даних зі списку в консоль, звільнення пам’яті тощо).
5. Значення елементів списку взяти самостійно такими, щоб можна було продемонструвати коректність роботи алгоритму програми. Введення значень елементів списку можна виконати довільним способом (випадкові числа, формування значень за формулою, введення з файлу чи з клавіатури).
6. Виконати над створеним списком дії, вказані за варіантом, та коректне звільнення пам'яті списку.
7. ***При виконанні заданих дій, виводу значень елементів та звільненні пам’яті списку вважати, що довжина списку (кількість елементів) <u>невідома</u> на момент виконання цих дій***. Тобто, не дозволяється зберігати довжину списку як константу, змінну чи додаткове поле.

При проєктуванні програм ***слід врахувати наступне*:**

1) при виконанні завдання кількість операцій (зокрема, операцій читання й запису) має бути мінімізованою, а також максимально мають використовуватися властивості списків;
2) повторювані частини алгоритму необхідно оформити у вигляді процедур або функцій (для створення, обробки, виведення та звільнення пам'яті списків) з передачею списку за допомогою параметра(ів).
3) у таких видів списків, як черга, стек, дек функції для роботи зі списком мають забезпечувати роботу зі списком, відповідну тому чи іншому виду списку (наприклад, не можна додавати нові елементи всередину черги);
4) програми мають бути написані мовою програмування С.

Варіант 17:

Ключами елементів списку є дійсні числа. Обчислити значення виразу: $a_1 \cdot a_n + a_2 \cdot a_{n-1} + \ldots + a_n \cdot a_1$, де $\boldsymbol{a_i}$ — $\boldsymbol{i}$-й елемент списку.

Текст програми:

```c
#include <stdio.h>
#include <stdlib.h>

typedef struct list_el {
   double data;
   struct list_el *next;
   struct list_el *prev;
} list_el;

list_el *create_list_el(double data) {
   list_el *new_list_el = (list_el *)malloc(sizeof(list_el));
   new_list_el->data = data;
   new_list_el->next = NULL;
   new_list_el->prev = NULL;
   return new_list_el;
}
```

```c
void insert_list_el(list_el **head, double data) {
    list_el *new_list_el = create_list_el(data);
    new_list_el->next = *head;
    new_list_el->prev = NULL;
    if (*head != NULL) {
        (*head)->prev = new_list_el;
    }
    *head = new_list_el;
}

list_el *get_last_list_el(list_el *head) {
    list_el *current = head;
    while (current != NULL && current->next != NULL) {
        current = current->next;
    }
    return current;
}

list_el *get_first_list_el(list_el *head) {
    list_el *current = head;
    while (current != NULL && current->prev != NULL) {
        current = current->prev;
    }
    return current;
}

//Calculate A1*An + A2*A(n-1) + ... + An*A1
double calculate(list_el *head, list_el *last, double S)
{
    if (head == NULL || last == NULL) return S;

    double head_data = head->data;
    double last_data = last->data;

    S += head_data * last_data;
    printf("a: %lf, b: %lf, S:%lf\n", head_data, last_data, S);
    return calculate(head->next, last->prev, S);
}

int main() {
    double S = 0;
    int n;
    printf("Enter the number of list_el: ");
    scanf("%d", &n);

    list_el *linked_list = NULL;

    printf("Enter the values of the list_els: ");
    for (int i = 0; i < n; i++) {
```

```
        double data;
        scanf("%lf", &data);
        insert_list_el(&linked_list, data);
    }

    list_el *head = get_first_list_el(linked_list);
    // Print created linked list
    printf("Linked list: ");
    while(head != NULL) {
        printf("%lf->", head->data);
        head = head->next;
    }
    printf("NULL\n");

    list_el *first_list_el = get_first_list_el(linked_list);
    list_el *last_list_el = get_last_list_el(linked_list);

    printf("Sum: %lf\n", calculate(first_list_el, last_list_el, S));

    while (linked_list != NULL) {
        list_el *current = linked_list;
        linked_list = linked_list->next;
        free(current);
    }
    return 0;
}
```

Тестування програми:

```
Linked list: 5.000000->4.000000->3.000000->2.000000->1.000000->NULL
a: 5.000000, b: 1.000000, S:5.000000
a: 4.000000, b: 2.000000, S:13.000000
a: 3.000000, b: 3.000000, S:22.000000
a: 2.000000, b: 4.000000, S:30.000000
a: 1.000000, b: 5.000000, S:35.000000
Sum: 35.000000
```

```
Linked list: 0.655345->-0.826997->5.112106->-7.369244->-9.999843->NULL
a: 0.655345, b: -9.999843, S:-6.553345
a: -0.826997, b: -7.369244, S:-0.458999
a: 5.112106, b: 5.112106, S:25.674633
a: -7.369244, b: -0.826997, S:31.768978
a: -9.999843, b: 0.655345, S:25.215634
Sum: 25.215634
```

```
Linked list: 22.000000->53.000000->46.000000->75.000000->14.000000->1.000000->NULL
a: 22.000000, b: 1.000000, S:22.000000
a: 53.000000, b: 14.000000, S:764.000000
a: 46.000000, b: 75.000000, S:4214.000000
a: 75.000000, b: 46.000000, S:7664.000000
a: 14.000000, b: 53.000000, S:8406.000000
a: 1.000000, b: 22.000000, S:8428.000000
Sum: 8428.000000
```

Висновок:

Засвоїв теоретичний матеріал лекцій набрався практичного досвіду в створенні та ефективному використанні двозв’язаних списків. Навчився створювати і працювати з процедурами і функціями для них. Набув досвіду у роботі з вказівниками на місця у пам'яті та звільнені самої пам'яті.